BRANCUSI, NEO-PLASTICISM, AND THE ART OF DESIGNING SPEECH-RECOGNITION APPLICATIONS

Blade Kotelly

SpeechWorks International, Inc.

ABSTRACT

Designing over-the-phone speech-recognition systems requires that the designer have a design methodology and philosophy that enables them to understand how to research, design, evaluate and re-design their application.

1. INTRODUCTION

While the technology behind speech-recognition applications is important, it doesn't matter how accurate or how robust the recognizer is, if the design of the application is poor. Speech-recognition systems that help people get flight information, make stock trades, or perform a variety of other tasks, are produced so that people, in real life, can accomplish these tasks. People who have no advanced training or instruction about how to use these systems are the ones who use most of the applications that are deployed. To design them well is an art. The *art of design* is embodied in all aspects of the design process –research, callflow design, audio file production, usability testing of the application, and most importantly, the designer's guiding philosophy.

2. BRANCUSI AS ARCHETYPE

Constantin Brancusi (1876–1957) was noted for his sculptures – particularly the shiny, metal and marble, streamlined-forms that represented animate objects in motion. A notable example, "Bird In Space" (1925), stands over 4 feet high – a representation of a hyper-streamlined bird launching upward toward the sky. And while this object is not obviously a bird complete with feathers and a beak, it does contain all the underpinnings necessary to create the effect in the viewers' mind of a bird moving through space with grace and energy. Brancusi said "They are imbeciles who call my work abstract; that which they call abstract is the most realist, because what is real is not the exterior form but the idea, the essence of things." [1]

When designing speech-recognition systems, the designer is obligated to know how the caller should feel *after* they've used the system. And the designer should know this before starting to design it. By working backwards (from understanding the user-experience to designing the callflow and finally writing the prompts), the designer will be sure to establish a philosophy that will keep the design consistent and that will create the desired effect for the user of the system.

3. DESIGNING

In order for designers to conceptualize this "final image" before beginning the design process, they need to understand the underpinnings of the design process.

The kind of speech recognition we're talking about is the kind that helps people when they call a company to find out some information or conduct a particular transaction; the design needs to help callers with their tasks. Basically, these systems ask questions, and then listen for the caller to answer them. The computer has a list of words or phrases it's listening for, and then tries to determine if the caller said one of those words or phrases or not. Now, keep in mind that the list of phrases that we're recognizing doesn't have to be small, but rather can be in the thousands or even in the hundreds of thousands. Some systems that were in production in the late 1990's could easily recognize people saying phrases to a stock-trading system, such as, "buy two hundred and twenty-five shares at a limit price of forty-seven and three-eighths." The design of these systems is both a science and an art (as is the design of almost everything) – we have particular methodologies that we follow which make it more like a science, but the way that a designer makes these systems feel, the way a designer casts and directs a voice talent, and the way a design brands the system considering the psychological component of the interaction, is much more of an art.

I've often asked people in various audiences if they've ever used a touch-tone system, and without fail, everyone says that they've used one. (For purposes of this paper I'll assume that the reader has used a touch-tone system before.) I then go on to ask the audience if they've ever used a poorly designed touch-tone system. I tend to also get a unanimous, affirmative response from the crowd. But when I ask if they've ever used a *good* touch-tone system, the percentage starts to drop. Often only 30% of the people in the room will tell me that they've ever used a good touch-tone system…but that notion gives me hope that there are indeed well designed touch-tone systems. So I then will ask if there are more good systems or bad ones. Everyone has always said that they are more bad systems. So

[1] (ca. 1957) Translated from "Propos de Brancusi" (collected by Claire Gilles Guilbert), Prisme des Arts (Paris), No. 12, May 1957, p. 6.

why are there more bad systems out there? When we examine the technology, we see that most touch-tone systems are very similar. There currently exist only 12 keys that the caller can press after they've heard a prompt played. That's it. In fact, the accuracy of these systems is something close to 99.999%. That's really good. So if the technology is basically the same between any two touch-tone systems; what makes a good one, or a bad one? I had to conclude that it's the design of the system that changes everything – not the technology.

What's interesting about speech systems is that they're new, and they're inherently social. Because of that, we need to make sure that when we design these systems, we take advantage of the knowledge of how humans communicate with each other, and how humans use machines to get tasks accomplished.

A good way to think about the process of designing is to use a standard 10-step design process that has been adapted to designing speech-recognition systems.

4. THE 10-STEP DESIGN PROCESS

The first steps deal with research. One should always start by "Understanding the *underlying* problem." If designers really understand the problem they're trying to solve, then they can approach it from the right angle. When I was young I used to ask my dad questions about a particular problem I had…but I often wouldn't put much context around the question. He could have answered the question right away, but he would usually ask me to explain *why* I was asking the question. Once I gave him the whole context, his answer would be much more useful to me – it was because he understood the underlying problem I was trying to solve, and not just the superficial aspects. Once we understand the underlying problem, we go on to an "Information Phase." In this phase, we look to find out what in the world exists that can inform us more about the current problem we're working on. You can think of it like this: if you were trying to learn how to cook, you might watch a cooking show, but you might also go to a restaurant to see how food is presented. You could even take a chemistry course to understand how chemical reactions and Bunsen burners work so that you could understand why yeast makes bread rise. For speech applications we like to explore how people accomplish the similar tasks in the real world to understand similar paradigms. After the information phase we go on to determine what consumers want in a formal "Consumer Research Phase." For example, if we were going to design a VCR we could interview several people, or send out questionnaires to determine the types of features we should include in a VCR. In speech systems we do the same types of things, so that we can determine the kinds of functionality that people *must* have, the types of functionality which would be considered nice-to-have, and even the functionality that wouldn't be perceived as very useful. This phase can be difficult to do, as the callers' desires may conflict with the desires of the client. A designer must be able to reconcile these two potentially competing ideas in order to create an application that solves both the business problems and the users' problems.

We then go on to conduct "Planned research." This is where we determine what type of things will be limiting the design process…things like budget limitations, software limitations, the number of people who might be assigned to the project, etc. This helps us to make sure that when we design the final system, we don't design something that can't be built. We then do a "Hazard analyses", to make sure that the design protects users against *unrecoverable* errors. error which the user can't rectify without a penalty. For example, if a system transfers the wrong amount of money between two bank accounts, the user should be able to transfer the money back – the error is recoverable. However, if a system allowed users to purchase stock without confirming the transaction first, then the user could be in financial ruin if the system mistakenly thought the user was indicating one company when in fact they wanted another.

Once all the research is done, the designer can start to be creative. The process begins with ideation – or brainstorming. From that the designer will come up with some concepts for the final design, and finally the designer will choose the best design to prototype.

The production of the software begins, and simultaneously the designer needs to start the audio production by casting a voice talent, and recording prompts and/or creating non-speech audio (such as music, or audio-icons).

The last formal phase comes after the design has been completely produced: "Verification of the Design." The designer must verify if the software works. What we mean by "works" is *not* quality-assurance testing to assure that the software is bug free, but rather, do people like using the application, and can they use it easily to get their tasks accomplished. If they can, the designers' work is done, but if not, the application needs to be redesigned and then tested again.

5. RESEARCH

Jacques Cousteau said that the best way to observe a fish is to become one. And we think that this is true about our customers and their callers. If we can get inside the head of the customer, then we can understand why they want to provide service using speech-recognition, and if we can get into the head of the caller, we can learn what it's like to *be* them. We start by researching the company.

5.1. Researching the company

There are many ways to analyze a company and the various issues that might influence the design of the speech-recognition system. One way to start is to understand the company's brand – are they a company that wants to project a strong, confident, almost stern image? Or do they have a casual, approachable image? We also want to understand the role of the proposed application. Will the system be the one product that represents the whole company – like a voice portal? Or is the system simply a single arm of the company, like a flight-information line for an airline? After we understand these elements, we research the caller.

5.2. Researching the caller

We need to learn a lot about callers, for example, basics like their demographics: where they live, how old they are, etc. But we also need to know how familiar they are with this task that will be automated. What language do they use to talk about this task? Is there particular jargon used when experienced users talk about their task?

Then we go on to see how they get this task accomplished now. Do they use the web or a touch-tone system? Perhaps they talk to a live operator, or interact with a person directly, like the interaction between a customer and a teller in a bank. We can use this information to see if there is anything we want the speech system to emulate – for example, the way a bank-teller would conduct a transaction. Finally, we need to make sure that we understand the callers' goals, know what's important to them, and know their mind-set when they are calling in. Are they happy to make this call (making a reservation for a resort) or are they doing something they wish they didn't have to do (registering a complaint about a defective product)?

There are lots of other types of research that are used as well: morphological analyses, call-center visits, technology briefings, etc. The important concept is that the research phase is used to allow the designer to familiarize themselves with the company, the callers, and the technology that will be used to implement the design. The technology-familiarization often extends beyond an understanding of the speech-recognition engine to be used, but often reaches into an understanding of the telephony environment. (Does the company capture ANI [caller-ID]? Do they have CTI to allow screen-pops to customer-service-representatives?) To paraphrase and modify a famous quotation: A problem well asked is a problem answered.

6. DESIGN

This is the point in the process where the designer gets to think about how the system should feel, as a whole object. The designer can quickly imagine a variety of methods that the system might work with the caller to accomplish particular tasks – they might have an idea of the right voice to use, and even know a good amount of text that might be spoken. But the designer does need to consider many things while fleshing out the idea.

Designers think about many aspects of the system from small ones to large over arching issues – for example they might think, "what kind of system will it be?" Now that idea can refer to a lot of things, so let's pick one example:

The designer might consider that certain systems are only touched once, meaning that the caller will only get one chance to use the system…let's compare that to a multi-touch system where we'd expect the average caller to use the system several times in their lives. The *one touch* system needs to convey all the ideas to the caller in that one call. A good example of this is an insurance quote – a caller might be a 52 year-old, male non-smoker. It is highly unlikely that the same caller would call the next day and be a 24 year-old, female smoker. So that kind of system needs to explain all the services of the insurance company in that one, single call. We can contrast this to a *multi-touch* system like a home-banking system, where we might expect callers to call in once or twice a month. Over time the caller would get to learn the system, and the system could then teach the caller how to get though the system faster for those things which the caller does often.

After considering issues like this, designers sit down with pen and paper – (well actually they usually stick to digital media) and they sketch out how the system will work – that is, the callflow. Usually the callflow starts with the beginning of the call, where the caller is welcomed, and then branches off into several directions based on which path the caller chooses. But then the designer needs to go further…the one thing that speech systems have – something that no other medium has in quite the same way – is "personality."

6.1. Personality

The personality of a system, in addition to reflecting the brand of the company, affects the callers' ability to understand the application, ability to learn how to use the application, and capacity to enjoy using it. Personality is expressed primarily by the text of the prompts, the voice talent that is cast, and the way that the voice talent is directed.

Here's an example of how a single voice talent can be directed to sound different for 2 different applications. In one example, the Hotel Information line, we might expect to hear a voice which is directed to sound "sales-y," that is, upbeat, engaged, excited about the product. In a contrasting example, we might expect to hear the same voice talent, directed differently, for a heavy-machinery information line designed to inform older users in the Midwest United States if a particular engine part is in or out of warranty. This application needs to come across as more to-the-point, and more straightforward.

In the first example, a voice talent could be directed so that the caller actually hears the smile in his voice. We want him to sound engaged, but not over the top (like a radio DJ).

In the second example, the voice might be directed to sounds "cooler" and to end each sentence with a paragraph-final feel to it. And there might not be much of a smile in his voice either. But in each example, the designer tries to create a personality that tries to bond with the user, without being insincere. The most important element of this interaction is the psychology behind the interaction.

6.2. Psychology

Experiments done by Cliff Nass and Byron Reeves at Stanford have shown that people treat computers the same way that they treat other people. Nass and Reeves tested lots of social psychology rules ("rules" meaning they've been tested thousands of times over the last hundred years) that apply between two people and substituted a computer for a person in the equation. Here's one example. We know that people are more polite to other people when they talk to them directly, rather than talking about them behind their backs. So the professors decided to see if that would be true about people using computers. Using two computers in the same room, all their subjects used one computer for a while. This computer would help the subjects with a particular task. Then half of the subjects were asked to stay at that computer, while the other half were asked to move to a different computer in that same room. All the subjects were then asked to open up a new program and evaluate how good the first computer they used was at helping them with their task. They found that all the people who evaluated the first computer on the other computer *in the same room* gave the first computer average scores, and the scores were all over the scale. But the people who used the first computer and evaluated it on the same computer gave it consistently higher scores in a narrower range – because they didn't want to offend this poor computer!

As it turns out, all the social psychology rules that normally apply between two people, which have been tested to date, also hold true for a person dealing with a computer. So why do designers of speech-recognition systems care about these findings? Well, they can use that knowledge about social interaction to establish close relationships…between the caller and the application. For example, a designer might employ psychological elements attributed to a teamwork relationship so that the caller feels like the system and they are *collaborating* on the task. Designers can also use a particular personality to give an identity-differentiation between similar products, for example, given two banks, one bank might be the stern bank, with a John Houseman-like voice, while another bank could sound more friendly, and relaxed. Both banks may allow their customers to conduct the same transactions, but to the caller, the systems might feel like very different banks. Also, designers can reduce churn in systems, that is, when people start to use a system – if they like the way it feels to use, they don't want to give up their interaction with it. The same way that people who like other people become friends – because they enjoy that interaction.

But to correctly support the psychological aspects of the interaction of the system, the designer needs to also understand the practical aspects of the design process. For example, just having a good understanding of the psychological aspects of how a system and person interact, doesn't mean the designer knows how best to give the user the information they need.

6.3. Nuts and bolts

The way that information is delivered is critical. The terminology that's used (whether we choose it to be technical or not), the discourse markers that we use when we're talking to other people (phrases like "Got it" or "Okay") and the way that we arrange the information are all critical to the success of the application. For example, I often ask people what the most important piece of information is to give to people when telling them about the status of a departing flight. They often say that the "time" is the most important piece of information. I disagree. I think that it's the "status." *If* it's on time, or *if* it's delayed. *That's* the most important piece of information. Often people know when the flight is supposed to leave, and if they hear that it's "on time", they're satisfied and can hang up if they choose.

United Airlines has a flight information system. The system is very social, and I believe that the social aspect of the interface enhances the understanding of the application. What often occurs is someone confirming an itinerary and then the system searching a database. If the system determines that the particular itinerary could lead to more than one possible flight, it then has to tell the caller about that situation, because the caller needs to do more work sorting through the data. Here's how that interaction works:

Speech-recognition system: "Okay, I'll look up flights that match that itinerary. Hold on…<database lookup>…I found a few flights that just about match that itinerary. (Three, to be exact.) Help me find the right flight. Here's the first one on my list."

I didn't want the computer to start by saying (after what could be a long database search…particularly if several hubs were closed due to bad weather), "Three flights appear to match your itinerary." The caller might immediately write down the number "3" without knowing why, or what relevance the number had to them. Instead I wanted to ease the caller into the idea that a few flights appear to almost match what they wanted.

Then the system informs the caller, almost parenthetically that there are "three" flights "to be exact". This gives the caller a sense about the scope of the remaining work without making them feel responsible to know that particular datum. Finally the last phrase is spoken quickly "*Here's the first one on my list*". Its spoken quickly to make the caller feel like the task won't be arduous. It's the same reason that when a young child is running around while playing soccer, then falls…they look up at you and think to themselves, "should I cry?" If you look at them with horror on your face, they cry; but if you say "Great way to get into the game! Let's get going," they're off and running and having a good time again. We don't want callers to hear that same statement spoken with a dour voice. We want them to feel like this next task will be a breeze.

And while there are a lot more elements that go in to the design of good speech-recognition systems, the basic idea is to use all the media (voice and sound) and an understanding of the social aspect of the interaction to optimize the design of the application. A good designer should be able to defend why every single word of an application is in the design.

Once the application is designed, it needs to be produced. Production generally involves coding and connecting to a back-end database, integrating with the telephony system, and the recording prompts. This produces a system that can be used by the intended audience. Or will it? In actuality, it's critical to make sure that the intended audience can use it. For the most part, these systems will never be used by people who will take time to learn how a particular complex interaction works, or who will be willing to train the system to understand their particular voice. These systems need to work out of the box. And the only way to achieve that level of elegance is to test it on an indicative set of users and then redesign the system accordingly to improve the design. I've used systems created in laboratories that don't recognize the word "Help" (while recognizing "I was thinking about taking this trip to Bermuda next month, do you think it's a good time of year to go?"). Real callers who need to accomplish tasks (like home banking, stock-trading, getting flight-information, etc.) need these systems to work well, and to help them out when they need it.

7. USABILITY TESTING

The goal of the usability test is to ensure that people like the system. Technically it evaluates a "conceptual model match" – which simply means: Do users get it? Do they understand how to use the application? Do they like it? We can also use the tests to make sure that the system recognizes the things that the callers will say. There are two main methods of testing. One is the "Wizard of Oz" method, where one person pretends to be the computer and one person pretends to be the caller. If it's done right, and the person who's pretending to be the computer sticks to the "script," then the designer can learn a lot of the potential problems before any coding or prompt recording takes place. Then once the application has been coded and the prompts are recorded, we then observe people using the system in a controlled space. The space is usually a usability laboratory, complete with a one-way mirror and a video camera. The subject performs some tasks and the designer observes them to see what problems they encounter. This way, before lots of real users get to the system, it's been modified and improved to ensure that it's already working right.

8. DESIGN PHILOSOPHY

But the most important thing to remember is that designers need to walk a mile in their callers' shoes. If a designer knows what it's like to be an actual caller – what it's like to be them when they're encountering problems and when they're problem-free, then they'll design a system which really works for their audience. Brancusi made objects that spoke to his intended audience in the way they allowed them to understand his ideas. He finely honed his objects until there existed just the right elements to convey the emotional and physical aspects – no more, no less. Designing great speech-recognition systems is done the same way. In fact, designing all great things is done the same way.

Can you imagine how much easier it would be to set the clock on your VCR if only someone asked their mom to try it out first? You'd bet that she'd tell them to go back to the drawing board.